IEGULDĪJUMS TAVĀ NĀKOTNĒ

Quantum finite automata on infinite words

Rūsiņš Freivalds

University of Latvia

European Social Fund project “Datorzinātnes pielietojumi un tās saiknes ar kvantu fiziku”
Nr.2009/0216/1DP/1.1.1.2.0/09/APIA/VIAA/044

The notion of finite automata working on an infinitely long tape was studied in much detail by B. A. Trakhtenbrot and R.Büchi around 1960. They found that, in spite of lack of immediate applications, this notion allows to describe the recognizable languages in terms of second order predicate logics.

Consider a language L_1 of infinite words in alphabet $\{0, 1, 2\}$:

$$L_1 = \{\text{the words containing infinitely many ones and only a finite number of zeros}\}$$

$L_1 = \{$the words containing infinitely many ones and only a finite number of zeros$\}$

The automaton has 3 states: q_0, q_1, q_2. If the word in the language L_1 then the state q_0 is entered only a finite nuber of times, and the state q_1 is enetered infinitely often.

We define sets $\{q_1, q_2\}$ and $\{q_1\}$ to be *accepting sets of states* and all the other sets of states to be *rejecting sets of states*.

Are there any languages recognizable by nondeterministic finite automata but not by deterministic finite automata?

NO

Are there any languages recognizable by nondeterministic finite automata with less states?

YES

Computational complexity began with the natural physical notions of time and space. Given a property, S , an important issue is the computational complexity of checking whether or not an input satisfies S.

For a long time, the notion of complexity referred to the time or space used in the computation. A mathematician might ask,
"What is the complexity of ***expressing*** the property S?"

It should not be surprising that these two questions - that of cheching and that of expressing - are related.

In early sixties Bűchi , Elgot and Trakhtenbrot showed how a logical formula may effectively be transformed into a finite state automaton accepting the language specified by the formula, and *vice versa*.
It demonstrates how to relate the specification of a system behaviour (the formula) to a possible implementation (the behaviour of an automaton) - which underlies modern checking tools.

The monadic second-order (MSO) logic of one successor is a logical framework that allows one to specify string properties using quantification over sets of positions in the string.

Now we consider an example how an automaton can be described by a formula.
Let the input word have the length n in the alphabet {a,b}. Then the considered sets are subsets of the set {1,2,…, n}.
$P_a(x)$ and $P_b(x)$ are, respectively, predicates

$P_a(x)$ = { the symbol number x in the input word equals a }
$P_b(x)$ = { the symbol number x in the input word equals b }

We use also individual predicates
S(x,y) = { y = x+1 }
first(x) = { x = 1 }
last(x) = { x = n }

We use in our example three set-variables having the following meaning:

X_1 = {all the positions i such that $i \equiv 1$ }

X_2 = {all the positions i such that $i \equiv 2$ }

X_0 = {all the positions i such that $i \equiv 0$ }

Now we wish to show how the regular language

{the length of the input word is a multiple of 3}

can be described. The MSO formula is as follows.

Now we wish to show how the regular language

{the length of the input word is a multiple of 3}

can be described. The MSO formula is as follows.

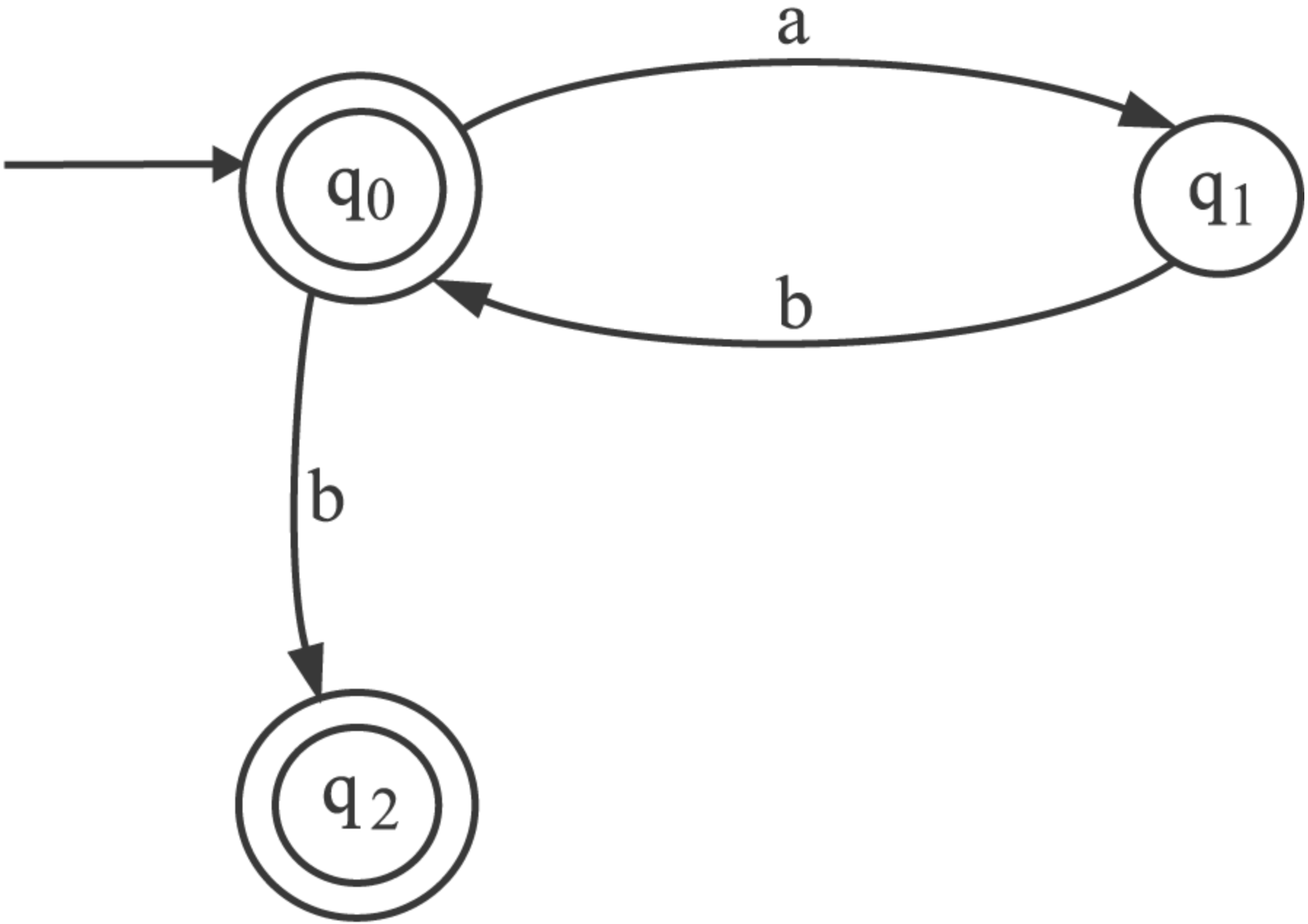
$$\exists\ X_1\ X_2\ X_0\ (\ (\forall x)\ (X_0(x) \vee X_1(x) \vee X_0(x)) \&$$
$$(\forall x)((\neg X_0(x) \& \neg X_1(x)) \vee (\ \neg X_0(x) \&\ \neg X_2(x)) \vee$$
$$(\ \neg X_1(x) \&\ \neg X_2(x))) \ \&\ \forall x\ (\text{first}\ (x) \rightarrow X_1(x)) \ \&\ \forall\ xy\ (S(x,y) \rightarrow$$
$$((X_1(x) \ \&\ X_2(x)) \vee (X_2(x) \ \&\ X_0(x)) \vee (X_0(x) \ \&\ X_1(x))) \ \&$$
$$\forall\ x\ (\text{last}(x) \rightarrow X_0(x)))$$

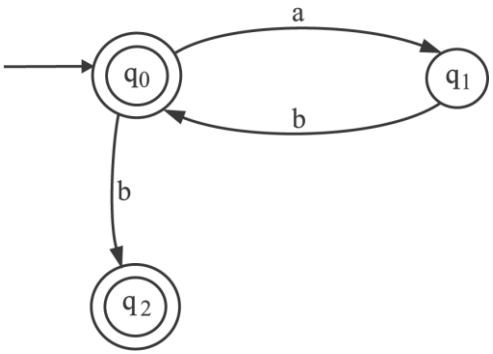
Now we wish to show how the regular language

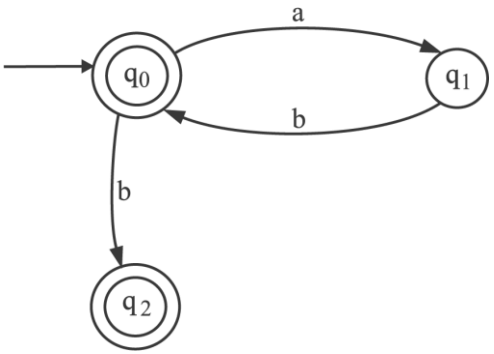
{the length of the input word is a multiple of 3}

can be described. The MSO formula is as follows.

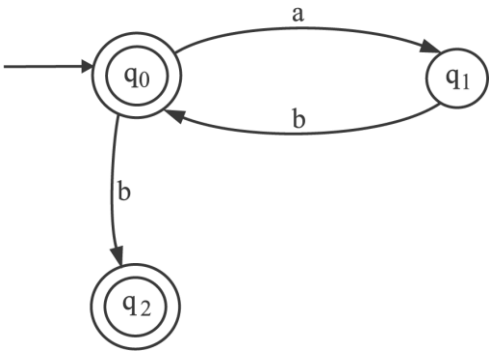
$$\exists\ X_1\ X_2\ X_0\ (\ (\forall x)\ (X_0(x) \vee X_1(x) \vee X_0(x)) \&$$
$$(\forall x)((\neg X_0(x) \& \neg X_1(x)) \vee (\ \neg X_0(x) \& \neg X_2(x)) \vee$$
$$(\ \neg X_1(x) \& \neg X_2(x))) \ \&\ \forall x\ (\text{first}\ (x) \rightarrow X_1(x)) \ \&\ \forall\ xy\ (S(x,y) \rightarrow$$
$$((X_1(x) \ \&\ X_2(x)) \vee (X_2(x) \ \&\ X_0(x)) \vee (X_0(x) \ \&\ X_1(x))) \ \&$$
$$\forall\ x\ (\text{last}(x) \rightarrow X_0(x)))$$


a
q0
q1
b
b
q2


a
q_0
q_1
b
b
q_2



Start = $(\forall x)\ (\forall y)((\text{first}\ (x)\ \&S(x,y)\&\ P_a(x)) \rightarrow X_1(y)) \vee ((\text{first}\ (x)\ \&\ S(x,y)\&\ P_b(x)) \rightarrow X_2(y))$



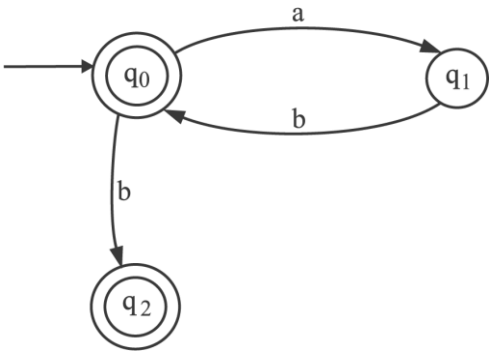
$$\text{Start} = (\forall x)\,(\forall y)((\text{first}(x) \,\&\, S(x,y) \,\&\, P_a(x)) \rightarrow X_1(y)) \vee ((\text{first}(x) \,\&\, S(x,y) \,\&\, P_b(x)) \rightarrow X_2(y))$$

$$\text{Transit} = (\forall x)\,(\forall y)(\, S(x,y) \rightarrow$$

$$((X_0(x) \,\&\, P_a(x) \,\&\, X_1(y)) \vee$$

$$\vee\, (X_0(x) \,\&\, P_b(x) \,\&\, X_2(y)) \vee$$

$$\vee\, (X_1(x) \,\&\, P_b(x) \,\&\, X_2(y)))$$



$$\text{Start} = (\forall x)\,(\forall y)((\text{first}(x) \,\&\, S(x,y) \,\&\, P_a(x)) \rightarrow X_1(y)) \vee ((\text{first}(x) \,\&\, S(x,y) \,\&\, P_b(x)) \rightarrow X_2(y))$$

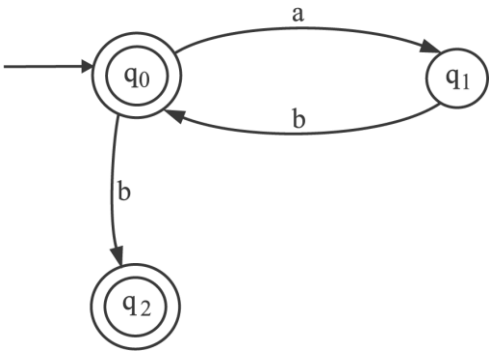
$$\text{Transit} = (\forall x)\,(\forall y)(\, S(x,y) \rightarrow$$

$$((X_0(x) \,\&\, P_a(x) \,\&\, X_1(y)) \vee$$

$$\vee\, (X_0(x) \,\&\, P_b(x) \,\&\, X_2(y)) \vee$$

$$\vee\, (X_1(x) \,\&\, P_b(x) \,\&\, X_2(y)))$$

$$\text{Accept} = (\forall x)(\, \text{last}(x) \rightarrow (X_0(x) \vee X_2(x))$$



Start = $(\forall x)\ (\forall y)((\text{first}(x) \&S(x,y)\& P_a(x)) \rightarrow X_1(y)) \vee ((\text{first}(x) \& S(x,y)\& P_b(x)) \rightarrow X_2(y))$

Transit = $(\forall x)\ (\forall y)(S(x,y) \rightarrow$

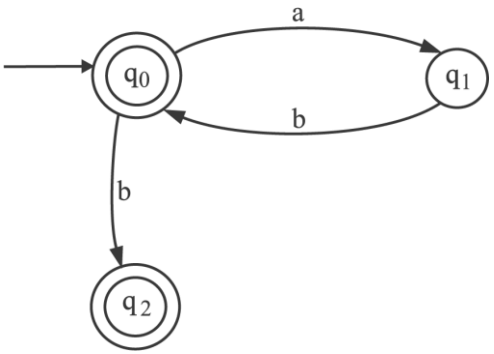
$((X_0(x) \& P_a(x) \& X_1(y)) \vee$

$\vee (X_0(x) \& P_b(x) \& X_2(y)) \vee$

$\vee (X_1(x) \& P_b(x) \& X_2(y)))$

Accept = $(\forall x)(\text{last}(x) \rightarrow (X_0(x) \vee X_2(x))$

FORMULA = $\exists\ X_1\ X_2\ X_0\ (\ (\forall x)\ (X_0(x) \vee X_1(x) \vee X_2(x)) \&$
$(\forall x)((\neg X_0(x) \& \neg X_1(x)) \vee (\ \neg X_0(x) \& \neg X_2(x)) \vee$
$(\ \neg X_0(x) \& \neg X_2(x))) \&$ START & TRANSIT & ACCEPT)



$$\text{Start} = (\forall x)\,(\forall y)((\text{first}(x) \,\&\, S(x,y) \,\&\, P_a(x)) \rightarrow X_1(y)) \vee ((\text{first}(x) \,\&\, S(x,y) \,\&\, P_b(x)) \rightarrow X_2(y))$$

$$\text{Transit} = (\forall x)\,(\forall y)(\, S(x,y) \rightarrow$$

$$((X_0(x) \,\&\, P_a(x) \,\&\, X_1(y)) \vee$$

$$\vee\, (X_0(x) \,\&\, P_b(x) \,\&\, X_2(y)) \vee$$

$$\vee\, (X_1(x) \,\&\, P_b(x) \,\&\, X_2(y)))$$

$$\text{Accept} = (\forall x)(\, \text{last}(x) \rightarrow (X_0(x) \vee X_2(x))$$

FORMULA $= \exists\ X_1\ X_2\ X_0\ (\ (\forall x)\,(X_0(x) \vee X_1(x) \vee X_2(x)) \,\&$

$(\forall x)((\neg X_0(x) \,\&\, \neg X_1(x)) \vee (\ \neg X_0(x) \,\&\, \neg X_2(x)) \vee$

$(\ \neg X_0(x) \,\&\, \neg X_2(x))) \,\&\, \text{START} \,\&\, \text{TRANSIT} \,\&\, \text{ACCEPT}\ )$

It needs to be reminded that Bűchi considers description of automata on infinite strings.

On the other hand, up to now quantum automata have been considered as processing finite words only. Perhaps there is some quantum machanics based motivation behind this restriction.

As for classical Bűchi automata, in the 1970's there was relatively little interest in these automata. There was some theoretical work on automata with infinite state spaces such as pushdown tree automata. However, the decision problems usually became undecidable. Thus, while of some theoretical interest, it did not appear to have major impact on Computing Science.

The situation changed on 1977 when Pnueli's paper appeared. Pnueli proposed the use of Temporal Logic for reasoning about continuously operating concurrent programs. Temporal Logic is a type of modal logic that provides a formalism for describing how the truth values of assertions vary over time. While there are a variety of different systems of Temporal Logic, typical temporal operators or modalities include Fp ("sometimes p") which is true now provided there is a future moment where p holds, and Gp ("always p”) which is true now provided that p holds at all future moments. As Pnueli argued, Temporal Logic seems particularly well-suited to describing correct behaviour of continouosly operating concurrent programs.

In 1974 Fagin gave a characterization of nondeterministic polynomial time (NP) as the set of properties expressible in *second-order existential logic*. Some the results arising from this approach include characterizing polynomial time (P) as the set of properties expressible in first-order logic plus a least fixed point operator, and showing that the set of first-order inductive definitions for finite structures is closed under complementation.

Theorem (Fagin 1974) (ESO) = NP

Example. Let the structure G=({1,2,…,n}, E) represent a graph of n vertices, and E be a single binary relation representing the edges of the graph. We say that the *graph G is 3-colourable* (in colors Red, Yellow, Blue) iff its vertices may be coloured with one of three colours such that no two adjacent vertices are the same colour.

Three colourability is an NP-complete property.

R,Y,B are set-variables expressing the set of the vertices coloured correspondingly.

$$(\exists R)(\exists Y)(\exists B)\ (\forall x)$$
$$(((R(x) \,\&\, \neg Y(x) \,\&\, \neg B(x)) \vee (\neg R(x) \,\&\, Y(x) \,\&\, \neg B(x)) \vee$$
$$\vee(\neg R(x) \,\&\, \neg Y(x) \,\&\, B(x))) \,\&$$
$$\&\ (\forall y)\ (\ E(x,y) \rightarrow ((\neg R(x) \,\&\, R(y)) \vee (\neg Y(x) \,\&\, Y(y)) \vee$$
$$(\neg B(x) \,\&\, B(y))))$$

R,Y,B are set-variables expressing the set of the vertices coloured correspondingly.

$$(\exists R)(\exists Y)(\exists B)\,(\forall x)$$
$$(((R(x) \,\&\, \neg Y(x) \,\&\, \neg B(x)) \vee (\neg R(x) \,\&\, Y(x) \,\&\, \neg B(x)) \vee$$
$$\vee(\neg R(x) \,\&\, \neg Y(x) \,\&\, B(x))) \,\&$$
$$\&\ (\forall y)\ (\ E(x,y) \rightarrow ((\neg R(x) \,\&\, R(y)) \vee (\neg Y(x) \,\&\, Y(y)) \vee$$
$$(\neg B(x) \,\&\, B(y))))$$

We now define (FO + LFP) to be the set of first-order inductive definitions. We do this by adding a least fixed point operator (LFP) to first-order logic.

Theorem (Immerman 1982, Vardi 1982) (FO + LFP) = P

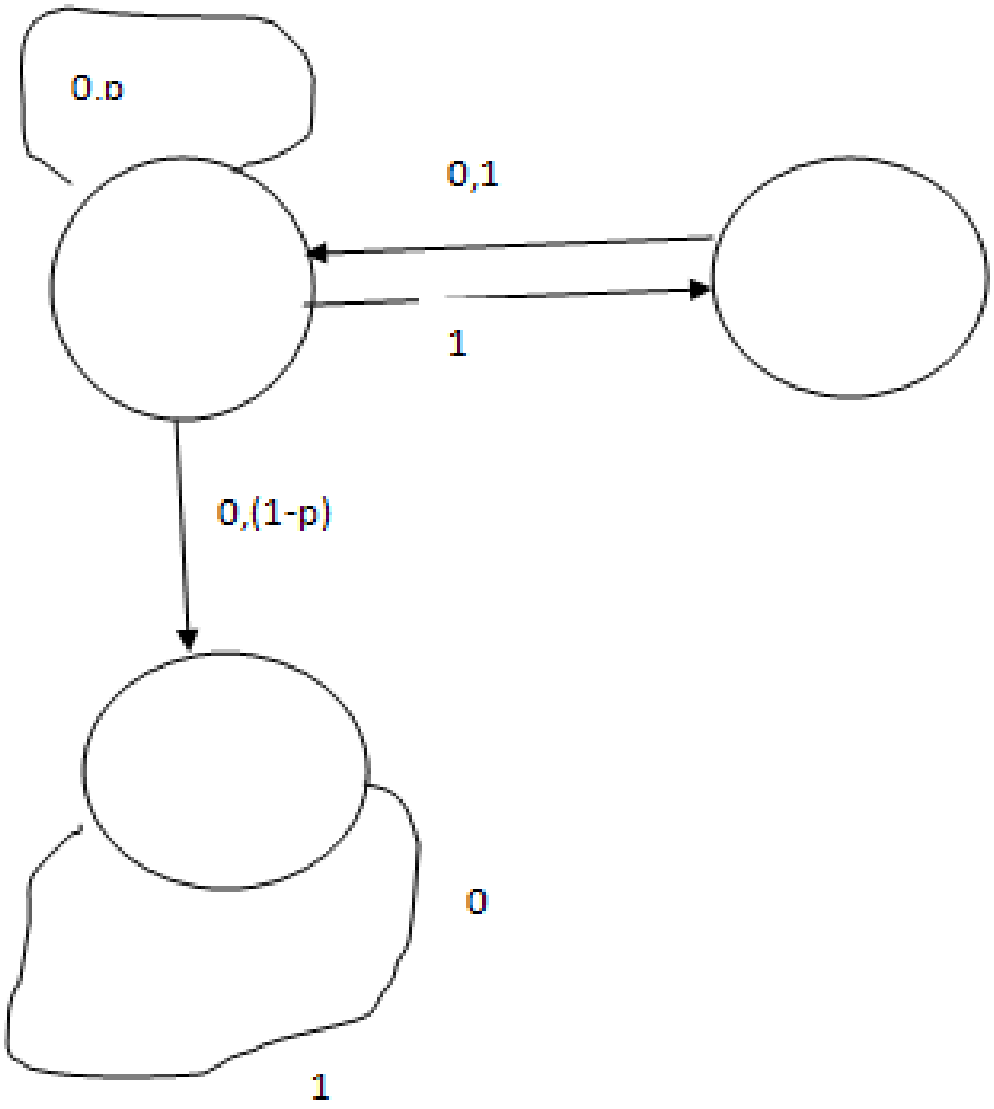
Theorem (Stockmeyer 1977) (SO) = PH

Theorem (Immerman 1982) PSPACE = $\bigcup_{k=1}^{\infty} \mathrm{FO}[2^{n^k}]$

This way, famous open problems in Theory of Computation turn out to be equivalent to purely logical problems.

For instance, P=?NP is equivalent to whether or not every second-order expressible property over finite ordered structures is already expressible in first-order logic using inductive definitions.

Are there any languages recognizable by probabilistic finite automata but not by deterministic finite automata?


0,p
0,1
1
0,(1-p)
0
1

Theorem. There exists a language of infinite words recognizable by a probabilistic finite automaton with probability 1 but not recognizable by deterministic finite automata.

Proof is based on **Borel-Cantelli lemma**.

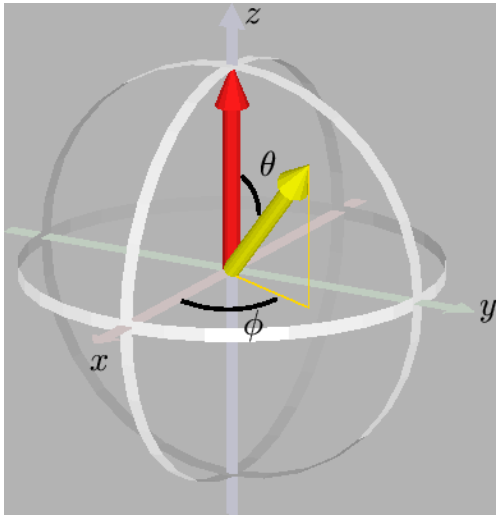
If $E_1, E_2, \ldots$ is a sequence of independent random events with probabilities $p_1, p_2, \ldots$ then:
-if $\sum_i p_i$ diverges then with probability 1 infinitely many of the events $E_1, E_2, \ldots$ have a positive outcome,
-if $\sum_i p_i$ converges then with probability 1 only a finite number of the events $E_1, E_2, \ldots$ has a positive outcome.

Unfortunately, it is not possible to use directly this construction to prove a counterpart of this theorem for quantum finite automata.

Qubits

- Quantum analogue of a classical bit
- Takes on values 0, 1, or superposition of states:

$$|\omega\rangle = \alpha|0\rangle + \beta|1\rangle \quad \textit{where} \quad |\alpha|^2 + |\beta|^2 = 1$$



$$|\omega\rangle = \cos(\theta / 2)\,|0\rangle + e^{i\varphi} \sin(\theta / 2)\,|1\rangle$$

Quantum finite automata (QFA) were introduced independently by Moore and Crutchfield (1997) and Kondacs and Watrous (1997). They differ in a seemingly small detail. The first definition allows the measurement only at the very end of the computation process. Hence the computation is performed on the quantum information only. The second definition allows the measurement at every step of the computation.

a

	q_1	q_2	q_3
q_1	2/3	2/3	-1/3
q_2	-1/3	2/3	2/3
q_3	2/3	-1/3	2/3

a

$$(1\ 0\ 0)\begin{pmatrix} 2/3 & 2/3 & -1/3 \\ -1/3 & 2/3 & 2/3 \\ 2/3 & -1/3 & 2/3 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

a

$$(1\ 0\ 0)\begin{pmatrix} 2/3 & 2/3 & -1/3 \\ -1/3 & 2/3 & 2/3 \\ 2/3 & -1/3 & 2/3 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

aa

$$(100)\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

a

$$(1\ 0\ 0)\begin{pmatrix} 2/3 & 2/3 & -1/3 \\ -1/3 & 2/3 & 2/3 \\ 2/3 & -1/3 & 2/3 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

aa

$$(100)\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$

The result of the matrix multiplication is the element shown in red color

These results repeat periodically:

2/3, 0, 2/3, 0, -1/3, 1, 2/3, 0, 2/3, 0, -1/3, 1, 2/3, 0, 2/3, 0, -1/3, 1,…

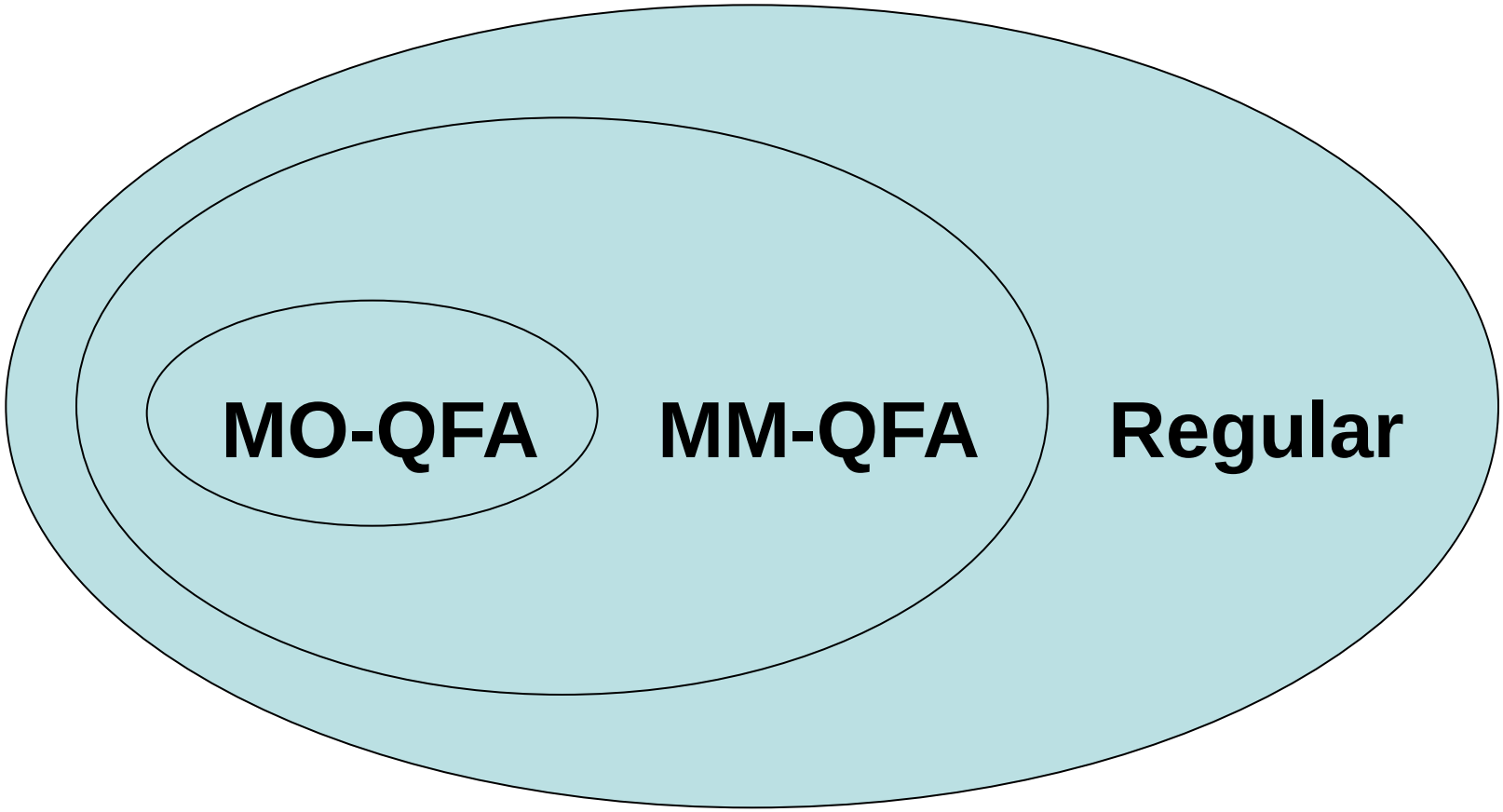
Hence we can consider this process as a recognition of whether the length of the input word is a multiple of 6. Only 3 states of the automaton! A deterministic automaton needs 6 states.

On the other hand, some probability of an error. The worst case error equals 4/9.

What is the class of languages recognized by QFAs?

What accepting probabilities can be achieved?

How does the size of QFAs (the number of states) compare to the size of deterministic (probabilistic) automata?


MO-QFA
MM-QFA
Regular

Theorem (Kondacs, Watrous, 1997).

1. All languages recognized by 1-way MM-QFAs are regular.

2. There is a regular language that cannot be recognized by a 1-way MM-QFA with probability $\frac{1}{2} + \varepsilon$ for any $\varepsilon > 0$.

$(0,1)^*1$

Theorem (Moore, Crutchfield 1997)

A language L is recognized by a MO-QFA if and only if L is a permutation language.

Example (Ambainis, Freivalds 1998) The language a^*b^*

The initial distribution $(\sqrt{(1-p)}, -\sqrt{p}, 0, 0)$

a

	q_0	q_1	q_{rej}	q_{acc}
q_0	$1-p$	$\sqrt{p(1-p)}$	$\sqrt{p}$	0
q_1	$\sqrt{p(1-p)}$	p	$-\sqrt{(1-p)}$	0
q_{rej}	$\sqrt{(1-p)}$	$-\sqrt{p}$	0	0
q_{acc}	0	0	0	1

Example (Ambainis, Freivalds 1998) The language a^*b^*

The initial distribution $(\sqrt{(1-p)}, -\sqrt{p}, 0, 0)$

	a				b			
	q_0	q_1	q_{rej}	q_{acc}				
q_0	$1-p$	$\sqrt{p(1-p)}$	$\sqrt{p}$	0	0	0	1	0
q_1	$\sqrt{p(1-p)}$	p	$-\sqrt{(1-p)}$	0	0	1	0	0
q_{rej}	$\sqrt{(1-p)}$	$-\sqrt{p}$	0	0	1	0	0	0
q_{acc}	0	0	0	1	0	0	0	1

Example (Ambainis, Freivalds 1998) The language a^*b^*

The initial distribution $(\sqrt{(1-p)}, -\sqrt{p}, 0, 0)$

a

	q_0	q_1	q_{rej}	q_{acc}
q_0	$1-p$	$\sqrt{p(1-p)}$	$\sqrt{p}$	0
q_1	$\sqrt{p(1-p)}$	p	$-\sqrt{(1-p)}$	0
q_{rej}	$\sqrt{(1-p)}$	$-\sqrt{p}$	0	0
q_{acc}	0	0	0	1

b

0	0	1	0
1	0	0	0
0	1	0	0
0	0	0	1

$

0	0	1	0
0	0	0	1
1	0	0	0
0	1	0	0

Theorem (Ambainis, Freivalds 1998) The MM-QFA described above recognizes the language a*b* with the probability $P = 0.68\ldots$ (where p is the solution of the equation $p^3 + p = 1$).

Theorem (Ambainis, Freivalds 1998) The MM-QFA described above recognizes the language a*b* with the probability $P = 0.68\ldots$ (where p is the solution of the equation $p^3 + p = 1$).

Theorem (Ambainis, Freivalds 1998) The language a*b* cannot be recognized by an MM-QFA with a probability exceeding 7/9.

Theorem (Ambainis, Freivalds 1998) The MM-QFA described above recognizes the language a*b* with the probability P = 0.68… (where p is the solution of the equation $p^3 + p = 1$).

Theorem (Ambainis, Freivalds 1998) The language a*b* cannot be recognized by an MM-QFA with a probability exceeding 7/9.

Theorem (Ambainis, Freivalds 1998) If a language can be recognized by an MM-QFA with a probability exceeding 7/9 then this language can be recognized by an MM-QFA with the probability 1.

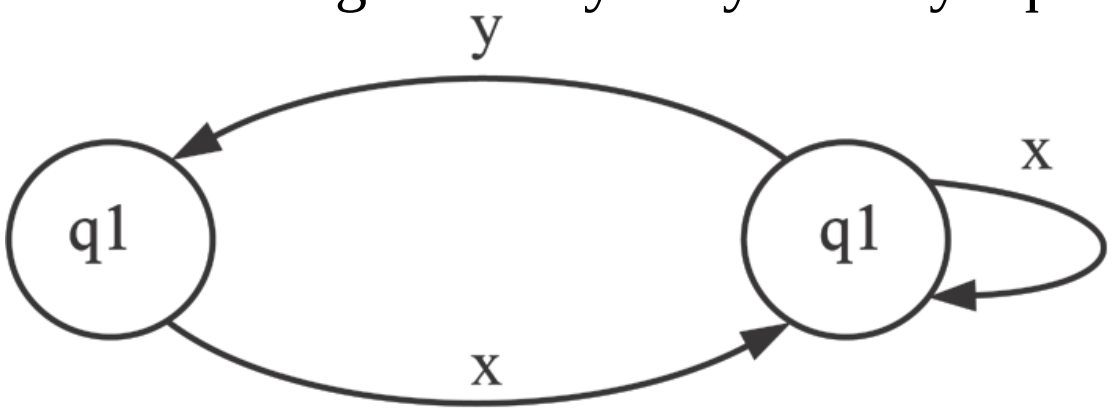
Theorem (Ambainis, Freivalds 1998)

1. For every prime p the language
L_p = { the length of the input word is a multiple of p}
can be recognized by a MO-QFA with no more than const log p states.
2. For every p arbitrary deterministic FA recognizing L_p needs at least p states.
3. For every p arbitrary probabilistic FA with a bounded error recognizing L_p needs at least p states.

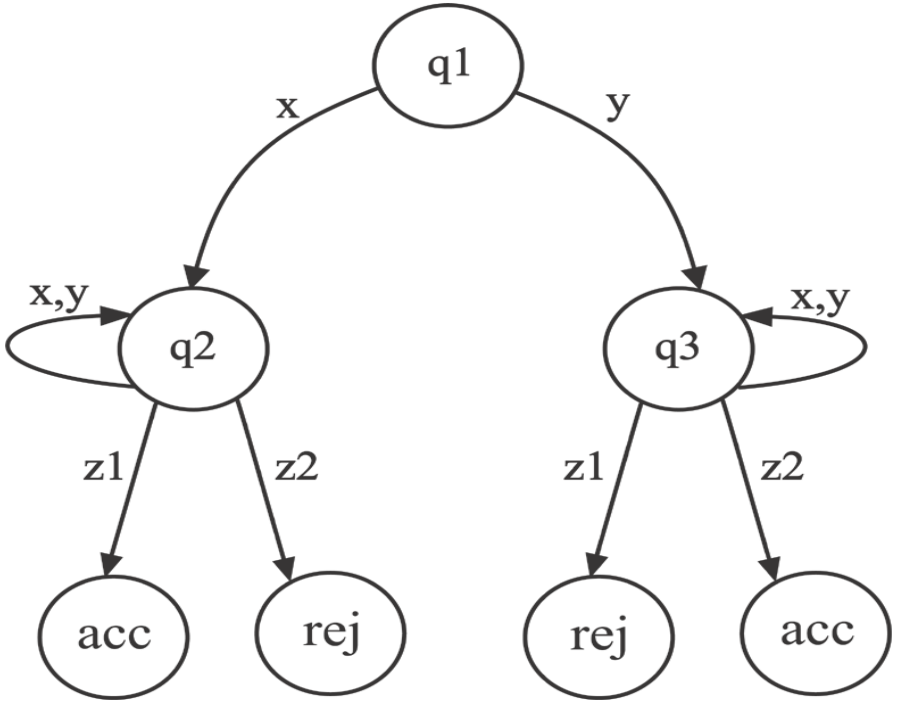
Theorem (Brodsky, Pippenger 1999) Let L be a language and M be its minimal automaton (the smallest DFA recognizing L). Assume that there is a word x such that M contains states q_1, q_2 satisfying:

1. $q_1 \neq q_2$
2. If M starts in the state q_1 and reads x,
it passes to q_2,
3. If M starts in the state q_2 and reads x,
it passes to q_1,
4. There is a word y such that if M starts in q_2 and reads y, it passes to q_1,

then L cannot be recognized by any 1-way quantum finite automaton.



Theorem (Ambainis, Ķikusts, Valdats 2001)
Let L be a language. Assume that there are words x, y, z_1, z_2 such that its minimal automaton M contains states q_1, q_2, q_3 satisfying:

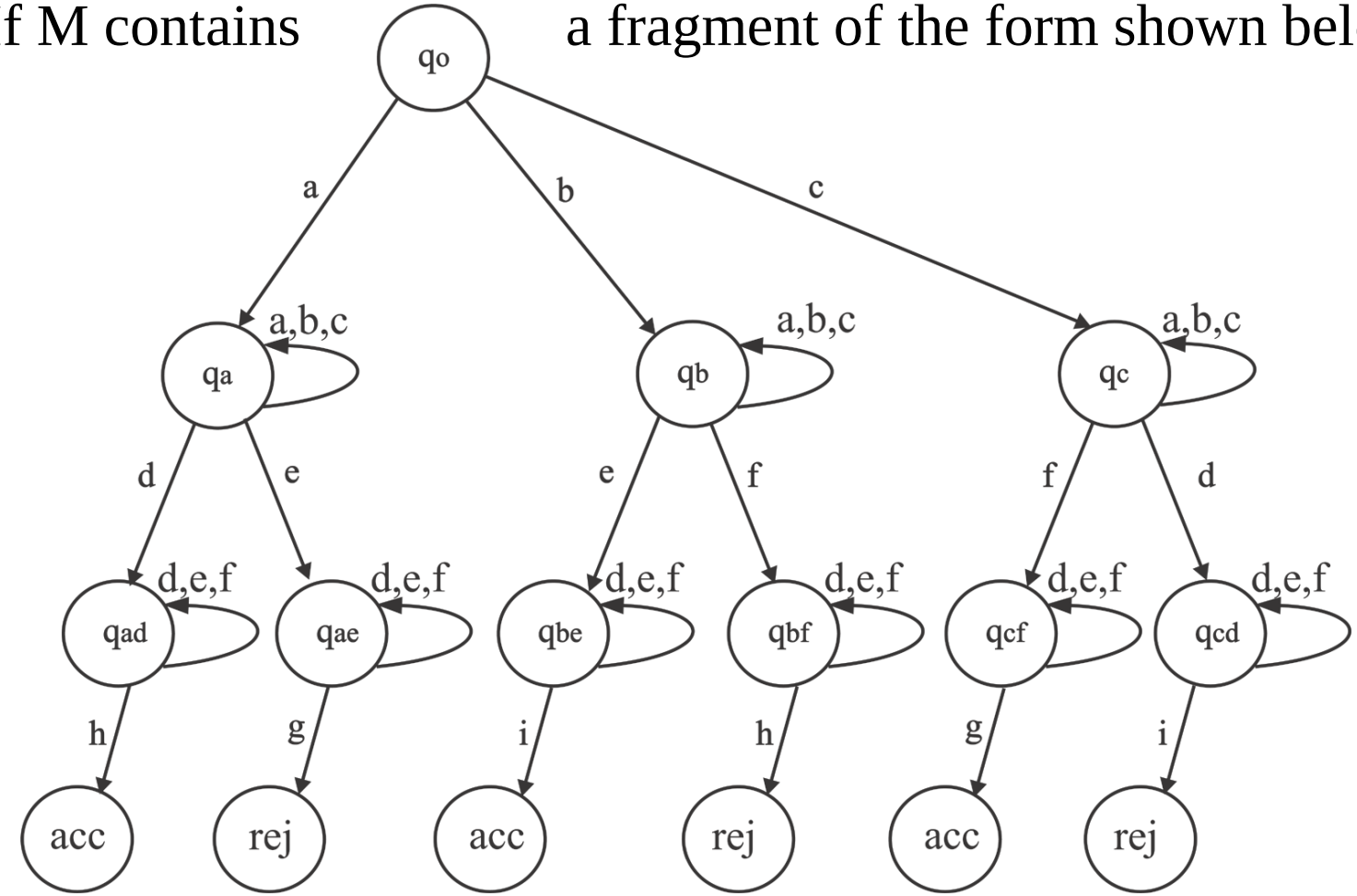


Then L cannot be recognized by any 1-way quantum finite automaton.

Theorem (Ambainis, Ķikusts, Valdats 2001)

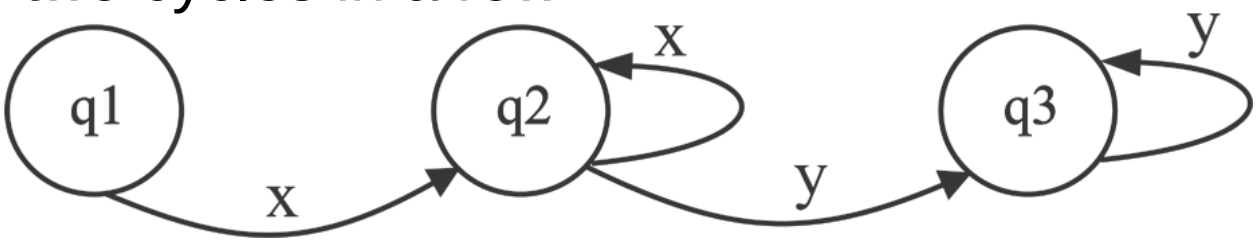
Let L be a language and M be its minimal automaton.

If M contains a fragment of the form shown below

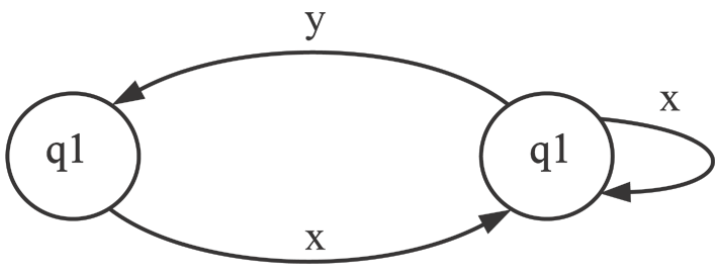


then L cannot be recognized by any 1-way MM-QFA.

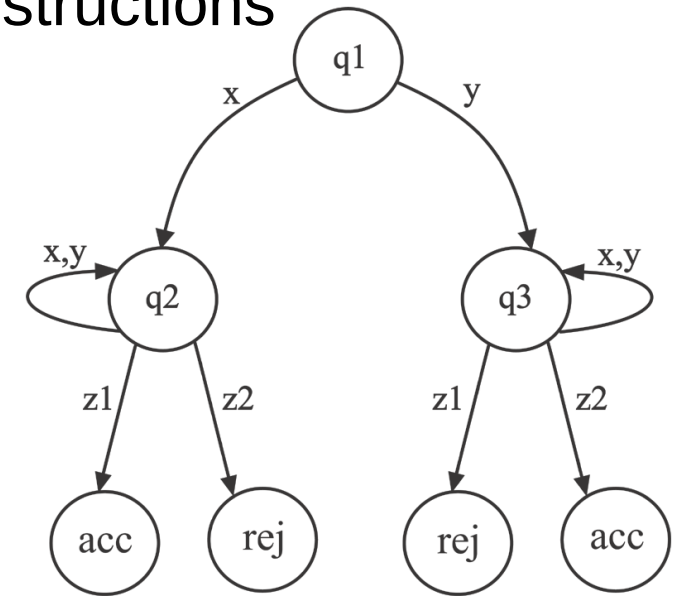
Theorem (Ambainis, Ķikusts, Valdats 2001) Let U be the class of languages whose minimal automaton does not contain "two cycles in a row"



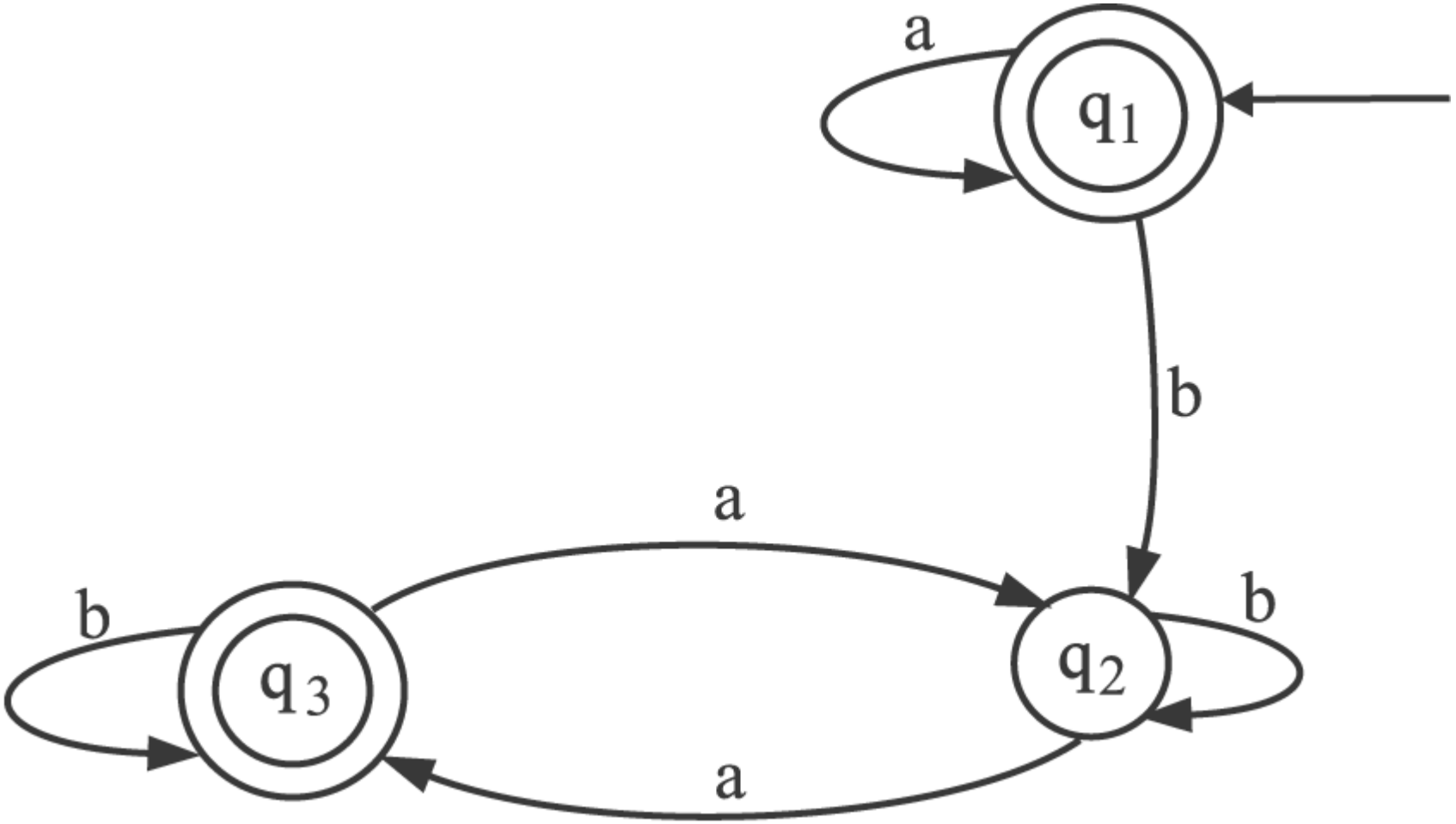
A language that belongs to U can be recognized by a 1-way MM-QFA if and only if its minimal deterministic automaton does not contain the "forbidden constructions"



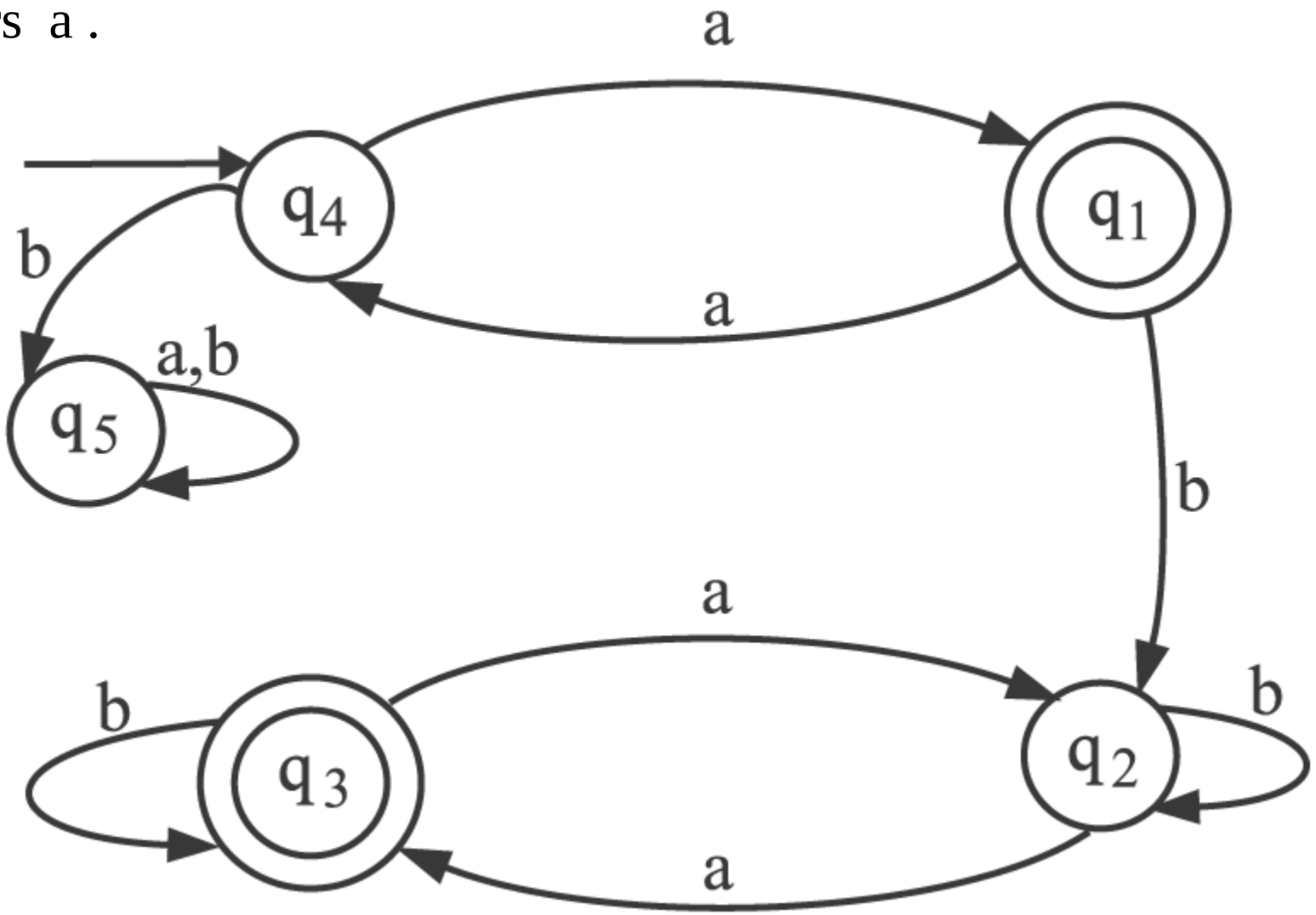
and



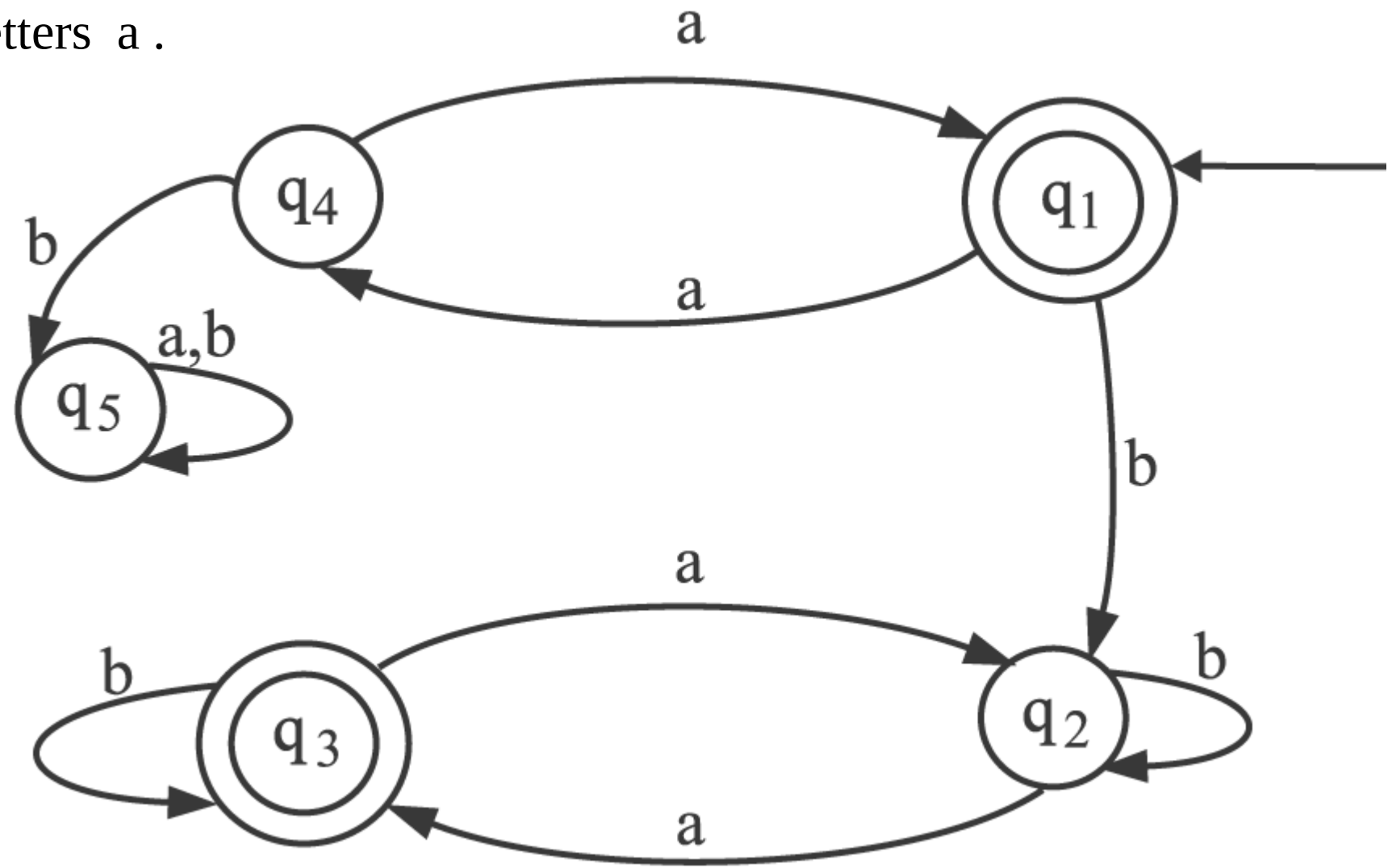
Let L_1 be the language consisting of all words that start with *any* number of letters a and after first letter b (if there is one) there is an odd number of letters a .



L_2 consists of all words which start with an *even* number of letters a and after first letter b (if there is one) there is an odd number of letters a .



L_3 consists of all words which start with an *odd* number of letters a and after first letter b (if there is one) there is an odd number of letters a .



$$V_\kappa = \begin{pmatrix} \sqrt{\frac{2}{3}} & \sqrt{\frac{1}{3}} & 0 & 0 & 0\,0\,0\,0 \\ \sqrt{\frac{1}{3}} & -\sqrt{\frac{2}{3}} & 0 & 0 & 0\,0\,0\,0 \\ 0 & 0 & -\sqrt{\frac{2}{3}} & \sqrt{\frac{1}{3}} & 0\,0\,0\,0 \\ 0 & 0 & \sqrt{\frac{2}{3}} & \sqrt{\frac{1}{3}} & 0\,0\,0\,0 \\ 0 & 0 & 0 & 0 & 1\,0\,0\,0 \\ 0 & 0 & 0 & 0 & 0\,1\,0\,0 \\ 0 & 0 & 0 & 0 & 0\,0\,1\,0 \\ 0 & 0 & 0 & 0 & 0\,0\,0\,1 \end{pmatrix}, V_a = \begin{pmatrix} 0\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0\,0\,0 \\ 1\,0\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1 \end{pmatrix}$$

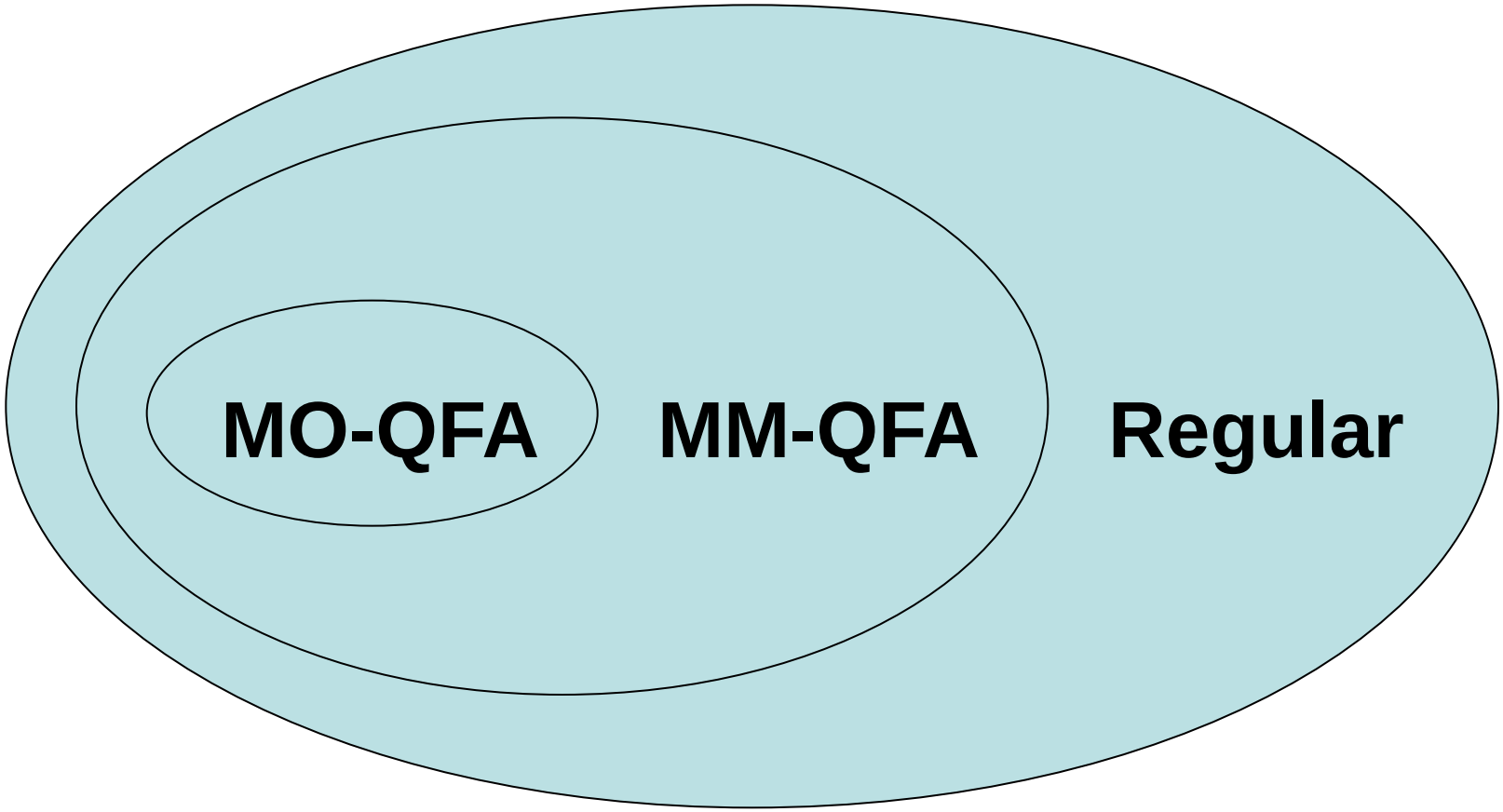
$$V_b = \begin{pmatrix} 0 & 0\,0\,0 & \sqrt{\frac{1}{2}} & \sqrt{\frac{1}{2}} & 0\,0 \\ 0 & 1\,0\,0 & 0 & 0 & 0\,0 \\ 0 & 0\,1\,0 & 0 & 0 & 0\,0 \\ 0 & 0\,0\,0 & 0 & 0 & 1\,0 \\ \sqrt{\frac{1}{2}} & 0\,0\,0 & \frac{1}{2} & -\frac{1}{2} & 0\,0 \\ \sqrt{\frac{1}{2}} & 0\,0\,0 & -\frac{1}{2} & \frac{1}{2} & 0\,0 \\ 0 & 0\,0\,1 & 0 & 0 & 0\,0 \\ 0 & 0\,0\,0 & 0 & 0 & 0\,1 \end{pmatrix}, V_\$ = \begin{pmatrix} 0\,0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,0\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,0 \\ 1\,0\,0\,0\,0\,0\,0\,0 \\ 0\,1\,0\,0\,0\,0\,0\,0 \\ 0\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,0\,0\,0\,0\,0 \end{pmatrix}$$

Theorem (Valdats 2000) There are two languages L_2 and L_3 which are recognizable by a MM-QFA but the union of them $L_1 = L_2 \cup L_3$ is not recognizable by a MM-QFA.

Theorem (Ambainis, Ķikusts, Valdats 2001) If the languages L_1 and L_2 are recognizable by a MM-QFA with probabilities p_1 and p_2 and $1/p_1 + 1/ p_2 < 3$ then $L_2 \cup L_3$ is also recognizable by QFA with probability $(2 p_1 . p_2)/(p_1 + p_2 + p_1 . p_2)$

Theorem (Ambainis, Ķikusts, Valdats 2001) If the languages L_1 and L_2 are recognizable by a MM-QFA with probabilities p_1 and p_2 and $p_1 > 2/3$ and $p_2 > 2/3$, then $L_2 \cup L_3$ is recognizable by MM-QFA with probability $p_3 > 1/2$.

There is no good description of the class of languages recognizable by quantum finite automata. Can one produce such a description in terms of a suitable logic?


MO-QFA
MM-QFA
Regular

For quantum automata on finite words two main definitions (and several variations of them) are used.

-MEASURE-ONCE (the measurement is performed only at the end of the computation)

-MEASURE-MANY (the measurement is performed at every step on special accepting and rejecting states)

However we need a correct definition for language recognition by quantum finite automata on infinite words.

We are to adapt the notion "the set of all states visited infinitely often". For quantum automata this is not so easy.

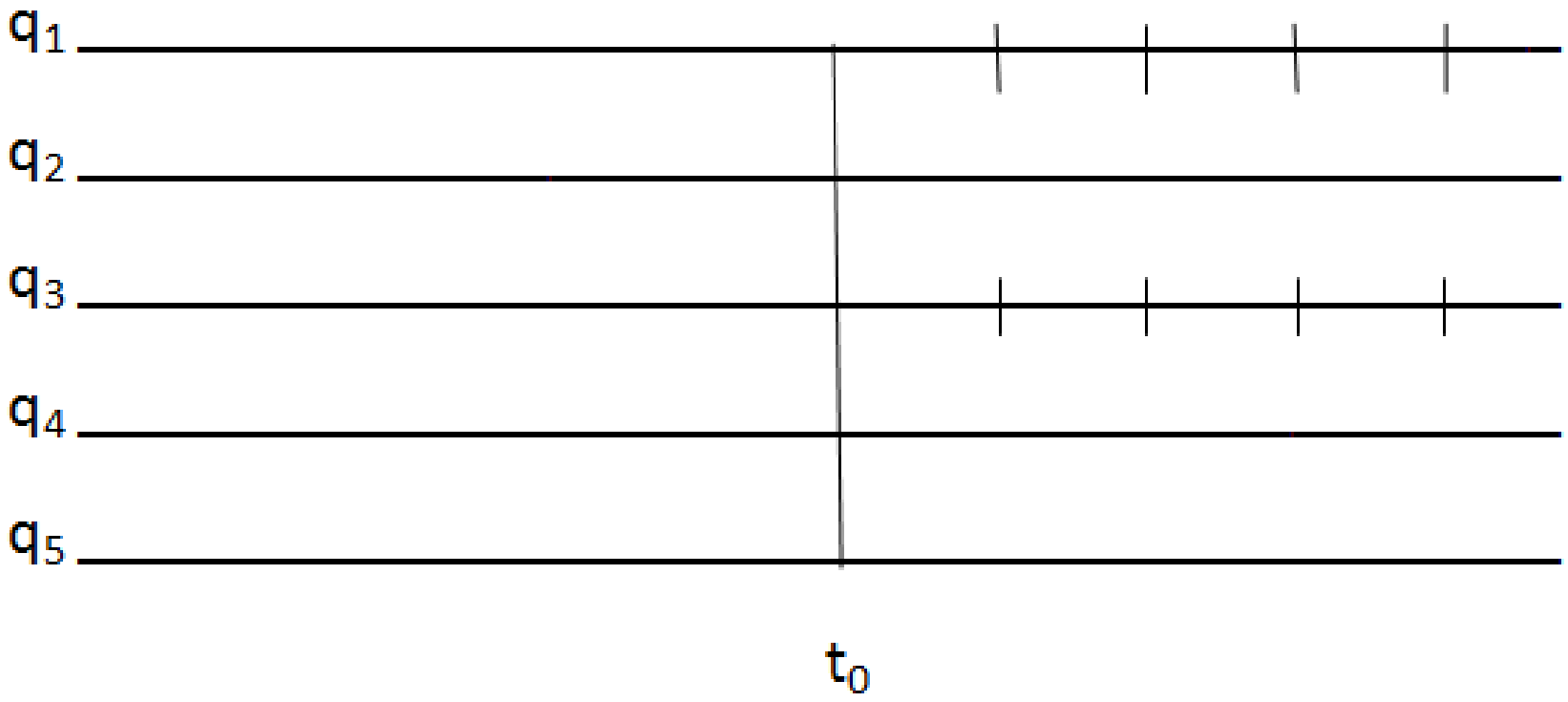
We will again have two definitions:

-MEASURE-MANY (the measurement is performed at every step on special accepting and rejecting states)

-MEASURE-ONCE (the measurement is never performed)

How to define:

"After the moment t_0 no state with exception of q_2, q_4, q_5 is entered"?


q_1
q_2
q_3
q_4
q_5
t_0

How to define:

"The probability of the event "After the moment t_0 no state with exception of q_2, q_4, q_5 is entered" "?

How to define:

"The probability of the event "No other states than q_2, q_4, q_5 are entered infinitely often" "?

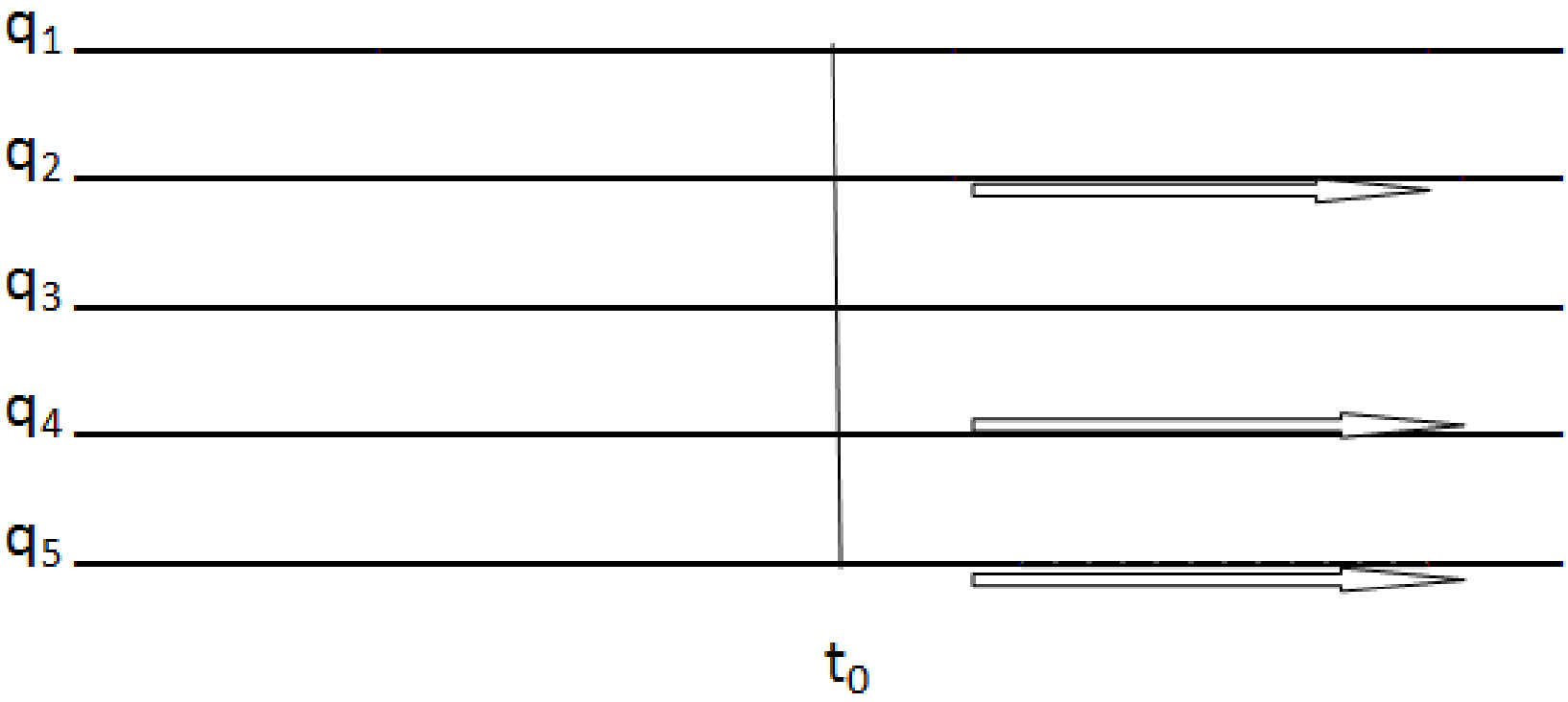
First of all, how to define a recognition of languages consisting of infinite words by quantum finite automata?

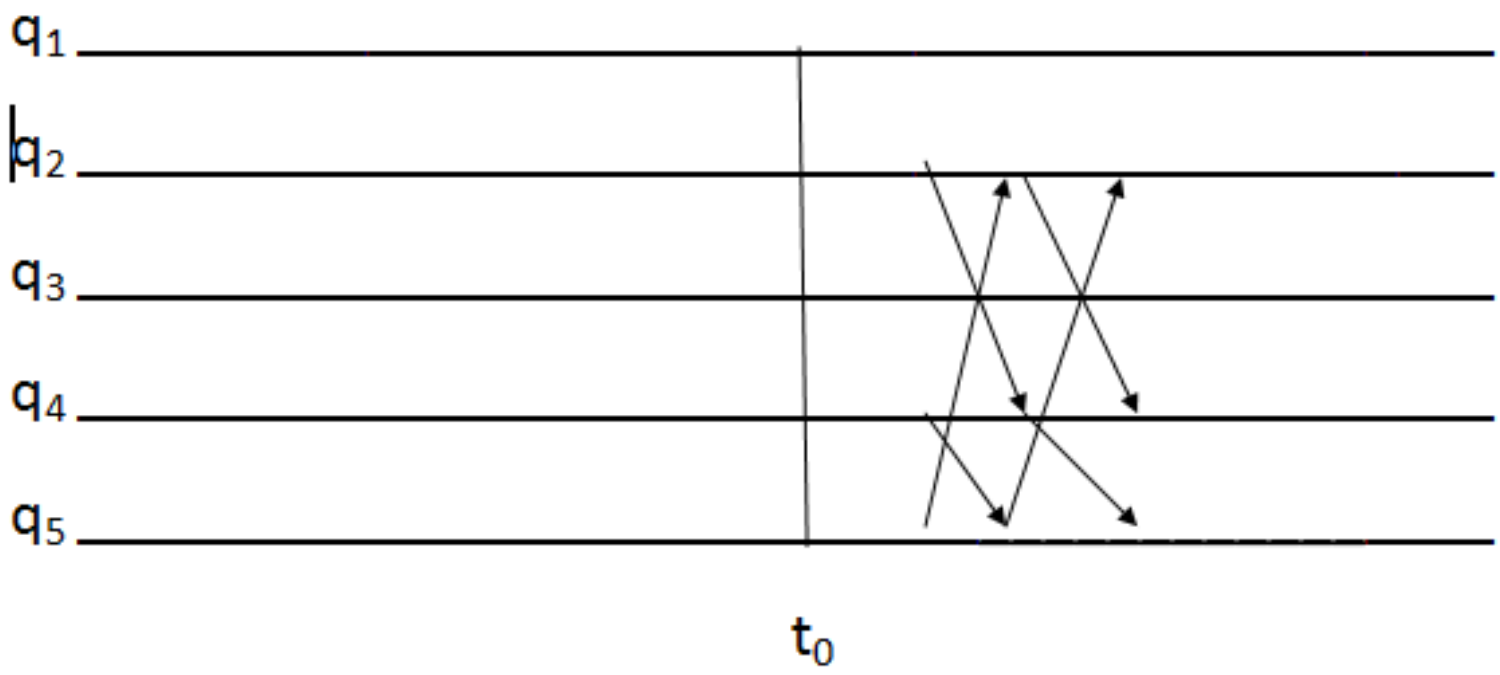
How to use measurement after an infinite number of steps of computation?

An answer is provided by ideas underlying the definition of Lesbegue measure.

However we need to define:

"The probability of the event "the states q_2, q_4, q_5 and only they are entered infinitely often." "


q_1
q_2
q_3
q_4
q_5
t_0


q_1
q_2
q_3
q_4
q_5
t_0

We define:

"The probability of the event: "After the moment t_0 the states other than q_2, q_4, q_5 are not entered and all the states q_2, q_4, q_5 are entered at least k times" "

We define:

"The probability of the event: "After the moment t_0 the states other than q_2, q_4, q_5 are not entered and all the states q_2, q_4, q_5 are entered infinitely often" "

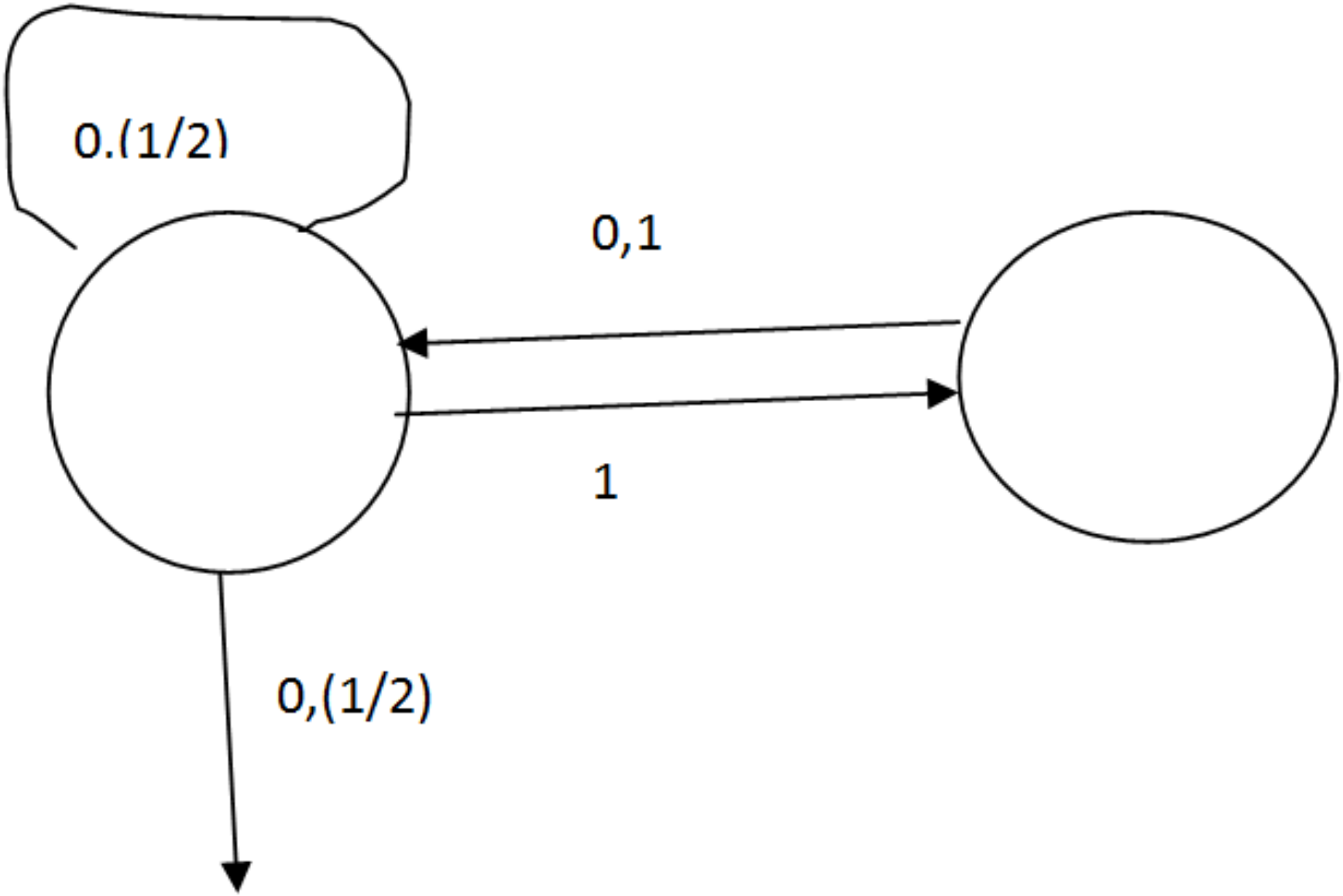
We define:

"The probability of the event: "The states other than q_2, q_4, q_5 are entered only a finite number of times and all the states q_2, q_4, q_5 are entered infinitely often" "

Finally we define:

"The probability of the event: "The set of states entered infinitely often is an accepting set of states" "

Theorem. There exists a language of infinite words recognizable by a MM-quantum finite automaton with probability 1 but not recognizable by deterministic finite automata.


0.(1/2)
0,1
1
0,(1/2)

Theorem. If a language of infinite words is recognizable by a MO-quantum finite automaton with a bounded error then it is recognizable by deterministic finite automata as well.

Thank you